IMPROVING SYMMETRIC STRUCTURE DURING RESOLUTION OF MOBILE FRIENDLY PROBLEMS IN WEB PAGES

Md. Aquib Azmain
BSSE 0705

A Thesis
Submitted to the Master of Science in Software Engineering Program Office
of the Institute of Information Technology, University of Dhaka
in Partial Fulfillment of the
Requirements for the Degree

MASTER OF SCIENCE IN SOFTWARE ENGINEERING

Institute of Information Technology
University of Dhaka
DHAKA, BANGLADESH

IMPROVING SYMMETRIC STRUCTURE DURING RESOLUTION OF MOBILE FRIENDLY PROBLEMS IN WEB PAGES

MD. AQUIB AZMAIN

Approved:

Signature	*Date*
Student: Md. Aquib Azmain	
Supervisor: Kishan Kumar Ganguly	

To *Makhluka Haque*, my mother
whose endless inspiration has always kept me motivated

Abstract

In software development, software usability is an important aspect to ensure the end-user does not strain or encounter problems with the use of a product or website's user interface. However, mobile friendly problem (MFP) causes the low quality of the website visibility and has a potential risk to decrease usability for a mobile user. The existing technique to solve mobile friendly problems does not incorporate with the symmetrical structure. To address this limitation, we have proposed an automatic repair technique that generates symmetric mobile friendly patches for a web page. This approach generates the fix by maximizing the balance score and minimizing the proportion score. The empirical evaluation shows that this approach gets better symmetric structure in 90.7% of the evaluated subjects. Moreover, A survey based evaluation shows that 88%, out of 54 subjects, have been considered as more preferable than the previous version by the users.

Acknowledgments

I would like to appreciate my supervisor Kishan Kumar Ganguly for his supportive guidance and continuous inspiration. His support has inspired me to relentlessly work for producing the best quality work I can throughout the project. I would also like to thank my classmates of BSSE 7th Batch for their continuous motivation throughout the project.

I am also thankful to the DSSE Student Research Group, Institute of Information Technology, University of Dhaka, for their continuous support and valuable feedback that helped me in improving my research work.

I am also thankful to Ministry of Information and Communication Technology, Government of the Peoples Republic of Bangladesh for granting me ICT Fellowship Tracking No 19FS11876, Date 04-09-2019.

Contents

List of Tables

List of Figures

Chapter 1

Introduction

The web is being accessed more and more on mobile devices. Mobiles have become a common medium of accessing internet [1, 2, 3, 4]. Designing websites to be mobile friendly ensures that the pages of that website perform well on all devices [5]. The websites which are developed for desktop viewport might be difficult to view and use on a mobile device. Alternatively, the mobile-friendly version is readable and immediately usable. Google has rephrased its ranking criteria and included mobile-friendliness as a criteria while returning search results for mobile devices in April 2015 [6]. This concludes that if a website is not mobile friendly, the probability of getting higher rank in the search results will be less to users. The main problem of existing approaches is violation of symmetric structure of the website. Symmetrical design relies on proportion to create a style with mirroring sides. Symmetry happens when the composition of design is distributed evenly around a central point or axis of a website. Symmetry is one of the Gestalt laws' most important predictors for appeal, and it is also an important factor in our aesthetic perception of websites [7, 8]. In Zheng et al.'s evaluation, symmetry significantly correlated with participants' subjective ratings on aesthetics [9]. n this chapter, the motivation behind this work and challenges have been discussed. This chapter also carries the research questions and contribution of the proposed self-adaptive system design mechanism. A section on how the report has been organized is mentioned at the end of this chapter.

1.1 Motivation

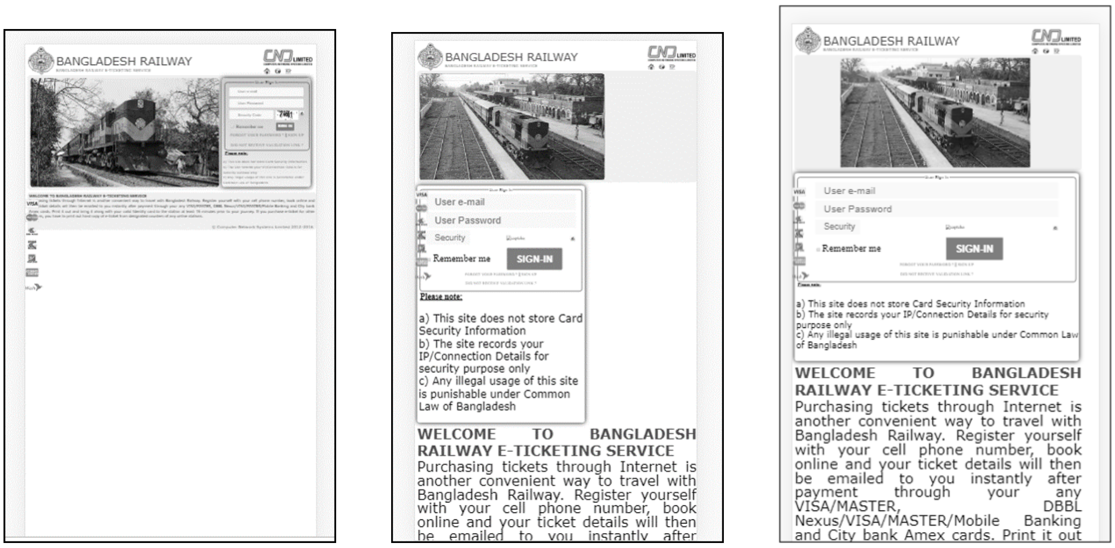
In software development, software usability is an important aspect to ensure the end-user does not strain or encounter problems with the use of a product or website's user interface. However, mobile friendly problem (MFP) causes the low quality of the website visibility and has a potential risk to decrease usability for a mobile user. So mobile friendly problems needs to be solved.

The work involved in making a mobile-friendly site can depend on developer re-sources, business model, and expertise. There are three manual processes to do this - 1) Responsive web design: Serves the same HTML code on the same URL regardless of the users' device (desktop, tablet, mobile, non-visual browser), but can render the dis-play differently based on the screen size. Responsive design is Google's recommended design pattern. 2) Dynamic serving: Uses the same URL regardless of device, but generates a different version of HTML for different device types based on what the server knows about the user's browser. 3) Separate URLs: Serves different code to each device, and on separate URLs. This configuration tries to detect the users' device, then redirects to the appropriate page using HTTP redirects along with the HTTP header. Existing approaches of automatic repair of mobile responsiveness issue are limited in detecting mobile friendly problems of a website so that the develeopers can repair them manually. For example, the tools for testing Mobile Friendliness developed by Google [5] and Bing [10], only helps to detect the MFP in a web page. Moreover, these approaches are time consuming and costly.

There have been some approaches to fix presentation issues of websites. For example, one approcah repairs layout cross browser issues caused by the inconsistencies in different browsers' rendering of a same web page. The other solution shows a framework of repairing faulty CSS of a web application but it does not address mobile friendly problems. Elsewhere. PhpRepair [11] repairs HTML syntax problems in web application. Moreover, Wang et al. [12] proposed an approach to fix presentation issues using static and dynamic analysis. However, none of these techniques identify mobile friendly problem and provide repairs for them. Mahajan et al. [13] proposed an approach to produce CSS patches automatically that can fix the mobile friendly problems of a web page [13].To distinguish the best fix proficiently, this methodology use different parts of the domain space to evaluate metrics identified by layout distortion and the computation are resembled to achieve the solution. None of the existing approaches ensures visual appeal during resolution of mobile friendly problem.

The goal of this research is to deliver a technique to improve symmetric structure of a website during resolution of mobile friendly problems. This proposed method will also be compared with the techniques in the existing works and it is expected that it will produce better results in all cases. It is also expected that this methodology will help to develop mobile friendly websites in a more efficient and convenient way. As mobile friendly are used for ensuring that web application increases its quality of usability, it is expected that the proposed approach will help to improve the overall quality of web application developed by industries. Moreover, this approach will decrease development time and cost as it automatically fixes CSS code according to the viewports of the mo-bile devices.

Moreover, mobile devices are the primary source of accessing the internet. Currently there is a large number of websites which have been developed exclusively for desktop computers. These websites, when accessed using mobile devices, perform very poorly. Even when websites are designed to be mobile-friendly, it adds a lot of development overhead. For example different stylesheets may be needed for different viewports. This pushes development cost, time and effort further



Please note:

a) This site does not store Card Security Information
b) The site records your IP/Connection Details for security purpose only
c) Any illegal usage of this site is punishable under Common Law of Bangladesh

WELCOME TO BANGLADESH RAILWAY E-TICKETING SERVICE
Purchasing tickets through Internet is another convenient way to travel with Bangladesh Railway. Register yourself with your cell phone number, book online and your ticket details will then be emailed to you instantly after

a) This site does not store Card Security Information
b) The site records your IP/Connection Details for security purpose only
c) Any illegal usage of this site is punishable under Common Law of Bangladesh

WELCOME TO BANGLADESH RAILWAY E-TICKETING SERVICE
Purchasing tickets through Internet is another convenient way to travel with Bangladesh Railway. Register yourself with your cell phone number, book online and your ticket details will then be emailed to you instantly after payment through your any VISA/MASTER, DBBL Nexus/VISA/MASTER/Mobile Banking and City bank Amex cards. Print it out

(a) Website with mobile friendly problems (b) Website after applying mobile friendly solution (c) Improved symmetric structure

Figure 1.1: Motivational Example

upward.

Improving mobile friendliness by improving readability of websites is just part of the equation. In order to retain users and alleviate dissatisfaction, visual appeal is the key. Our proposed approach will not only solve mobile friendly problem but also improve visual appeal of a website. So, this approach aims to be a practical low cost solution to improve their mobile friendliness with little or no additional development effort.

There is a motivational example(1.1) given below. 1.1a shows a website with various mobile friendly problems. After applying the solution for mobile friendly problems, the website has asymmetric structure (1.1b). By this example it shows that symmetric structure is needed to maintain visual aesthetic. 1.1c is the final result that shows how the website looks after fixing the structural problem.

1.2 Research Questions

Ensuring symmetric structure along with mobile friendliness is crucial for a website. The existing approaches focus on the solution of mobile friendly problems.

However, the visual appeal is ignored during resolution of mobile friendly problems. All these lead to the following research question.

1. How to improve symmetric structure during resolution of mobile friendly problems which are font sizing, tap sizing and content sizing of a web page?

   By answering the following sub-questions this question can be answered.

   (a) How to identify CSS properties and values of the HTML elements that cause asymmetry of a web page?
   By answering the question, we will get an approach to find some problematic HTML elements that cause asymmetry. Moreover, the CSS properties and the values of the target elements will be identified. For example, an image containing absolute positioning, fixed height and fixed length can cause asymmetry. To detect the asymmetry, the approach will check the balance between the two sides of a web page according to the centralized vertical axis. The approach needs to identify all these problematic properties and values that may cause asymmetry.

   (b) How to generate patches that improve symmetric structure maintaining trade-off between symmetry and layout distortion in different viewports?
   By answering the question, we will get an approach to generate patch that will modify some CSS properties like position, height, width, padding etc. so that the whole page will be symmetrically structured. However, improving symmetry can cause layout distortion. Layout distortion means overlapping of the segments of a website. Therefore, the approach will generate patches that will maintain trade-off between symmetry and layout distortion. For the aforementioned ex-ample, the approach will update the position property value (static, relative or

sticky) to achieve the most symmetric structure possible for that web page. However, the updated element can overlap on another elements that cause segment overlapping. To resolve this, the approach will re-position the overlapped segments maintaining symmetry.

1.3 Contribution and Achievement

This thesis proposes an repair technique that generates symmetric mobile friendly patches for a web page. For determine symmetry this approach introduces two symmetry based metrics named balance score and proportion score. Using the values of these metrics in a search based approach, this technique generate candidate fixes for the website. The best fix is generated by maximizing the balance score and minimizing the proportion score.

To evaluate the solution of symmetric mobile friendly approach, I used two different ways. The first one is the metric method. The repaired version has achieved better score in terms of balance score and proportional score. Therefore the effectiveness of the approach on the basis of balance is 90.7%. Moreover, 52 out of 54 subjects have lost proportion value. The effectiveness on the basis of proportion score is 96.2%. It confirms that the repaired versions have performed better than the previous mobile friendly technique. Moreover, A survey based evaluation shows that 88% subjects, out of 54, have been considered as more preferable than the previous version by the users

1.4 Organization of the Report

In this section, the organization of the report has been shown to provide a roadmap to this document. The organization of the chapters in this report has been mentioned in the followings.

Chapter 2: The definitions and background information of improving symmetric structure during resolution of MFP in this chapter.

Chapter 3: In this chapter, the existing works for improving symmetric structure during resolution of MFP have been presented is a structural way.

Chapter 4: This chapter contains the proposed methodology for the system of improving symmetric structure during resolution of MFP.

Chapter 5: The experimental setup, implementation and result analysis based on a case study have been provided in this chapter.

Chapter 6: This is the chapter that summarizes the whole report. Lastly, the chapter concludes the thesis with considerable future remarks.

Chapter 2

Background Study

This chapter provides background knowledge that is necessary throughout the approach. Section 2.1 discusses the basics of a generic web app and the rendering mechanism of user interface in a web page. This section also provides information about HTML, CSS and document object model. In section 2.2, I discuss types of mobile friendly problems occur in web pages. Besides, this section shows the difference between the viewports of mobile and desktop. In Section 2.3, I discuss the meaning and importance of symmetry in web apps.

2.1 Web App Fundamentals

A web application is a software application involved with client and server. The main code logic exists on a server remotely and the client can run the web application through a web browser through internet. Basically, the web applications have a three-tier architecture [14, 15]. These are:

1. Presentation tier: This tier displays the web application to the users. Their browsers render the Hyper Text Markup Language (HTML).

2. Business logic tier: This tier holds the core responsibility of the web application. Besides, the business logic tier determines how the other tiers interact

with one another.

3. Data tier: The data tier hosts servers where all the information is stored.

The end users only interact with the presentation tier and submits Hyper Text Transfer Protocol (HTTP) message (request) to business tier [16]. After processing, business logic tier interacts with data tier and generates related resources and responses back to the client which is rendered as HTML page to the browser. This architecture is given in image 2.1.

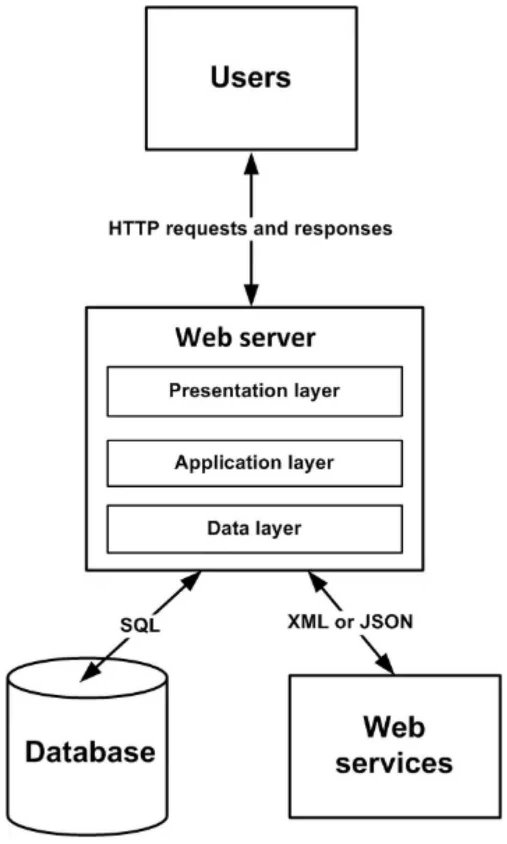


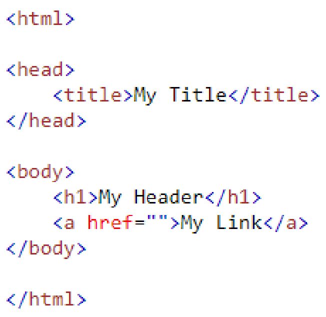
Figure 2.1: Three Tire Architecture

Model-view-controller(MVC) design is followed by modern web applications. The application runs on a server through the internet and produces the view of the web pages to a the client browser. My solution approach focus on repairing the display part of a web application.

2.1.1 Document Object Model

The Document Object Model (DOM) is a language-independent interface that constructs an XML or HTML document as a tree structure where in each node is an object which is a part of the document. To render an HTML page, the rendering engine parses the HTML code to construct the DOM tree. A tokenization algorithm is used to parse the HTML code. It parses the code as start tags, end tags and attributes. Every HTML element for example, html, body, H1 and a, is added to the Document object model tree. Every element has a unique reference in DOM. This reference is names an "XPath". After the document object tree is created, the subsequent step is render-tree development. The render-tree shows to the visual order where the HTML element will be shown. The render-tree is developed by parsing the CSS that holds the visual attributes of a web page. The next step is to choose the layout of the render tree. The render-tree is mapped by the DOM tree. A flow based model computes the bounding box for every element that is to rendered. After that, the render tree is printed. [17, 18].

The figure 2.2 shows an example how Document object model (figure 2.2b) is generated from an HTML code (figure 2.2a).



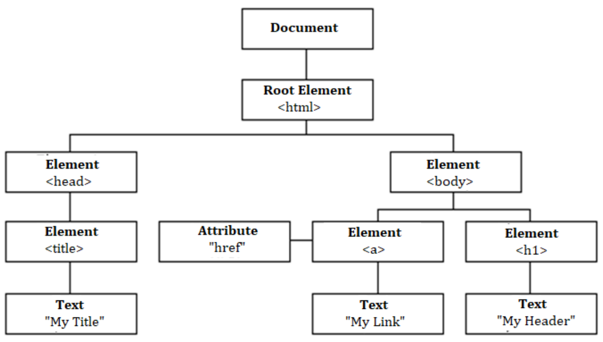
```
<html>

<head>
    <title>My Title</title>
</head>

<body>
    <h1>My Header</h1>
    <a href="">My Link</a>
</body>

</html>
```

(a) HTML Code



(b) Generated DOM Tree

Figure 2.2: DOM Example

2.2 Mobile Friendly Problems

Numerous sites are not intended to smoothly deal with clients who are visiting to their web pages through a cell phone or tablet. These risky websites may cause a number of usability issues, for example, cluttered navigation, unreadable text, or content that causes overflow to the device's viewport. Therefore, the user needs to use zoom and pan to view the web contents properly. These issues are altogether mentioned as Mobile friendly problems (MFPs) and can cause a low user experience. There are several types of mobile friendly problems such as [19]:

1. Uses incompatible plugins: The page includes plugins, such as Flash, that are not supported by most mobile browsers We recommend redesigning your page using modern, broadly-supported web technologies, such as HTML5.

2. Viewport not set: The page does not define a viewport property, which tells browsers how to adjust the page's dimension and scaling to suit the screen size. Because visitors to your site use a variety of devices with varying screen sizes—from large desktop monitors, to tablets and small smartphones—your pages should specify a viewport using the meta viewport tag.

3. Viewport not set to "device-width": The page defines a fixed-width viewport property, which means that it can't adjust for different screen sizes. To fix this error, adopt a responsive design for your site's pages, and set the viewport to match the device's width and scale accordingly.

4. Content wider than screen: Horizontal scrolling is necessary to see words and images on the page. This happens when pages use absolute values in CSS declarations, or use images designed to look best at a specific browser width (such as 980px). To fix this error, make sure the pages use relative width and position values for CSS elements, and make sure images can scale as well.

5. Text too small to read: The font size for the page is too small to be legible and would require mobile visitors to “pinch to zoom” in order to read. After specifying a viewport for your web pages, set your font sizes to scale properly within the viewport.

6. Clickable elements too close together: Touch elements, such as buttons and navigational links, are so close to each other that a mobile user cannot easily tap a desired element with their finger without also tapping a neighboring element. To fix these errors, make sure to correctly size and space buttons and navigational links to be suitable for your mobile visitors.

2.2.1 Mobile vs Desktop Viewports

Web applications are accessed through mobile devices more in recent years. As of 2019, 40 percent of the time spent on websites come from mobile devices which was near zero percent in 2007 [20]. The problem is the viewport of a web page is not the same for both mobile and desktop. A viewport is a polygon viewing region in computer graphics [21]. In computer graphics theory, there are two region-like notions of relevance when rendering some objects to an image. In textbook terminology, the world coordinate window is the area of interest (meaning what the user wants to visualize) in some application-specific coordinates, e.g. miles, centimeters etc. The viewport is the user’s visible area of a web page [22]. The viewport varies with the device, and will be smaller on a mobile phone than on a computer screen. Before tablets and mobile phones, web pages were designed only for computer screens, and it was common for web pages to have a static design and a fixed size.

There are 3 major types of viewports: 1) Desktop, 2) Smartphones, 3)Tablets. All three have different range of display heights and widths. Maximum width of desktop screen is considered as 991px, 767px for tablets and 479px for smartphones

[23]. The visual difference between these viewports are shown in figure 2.3.

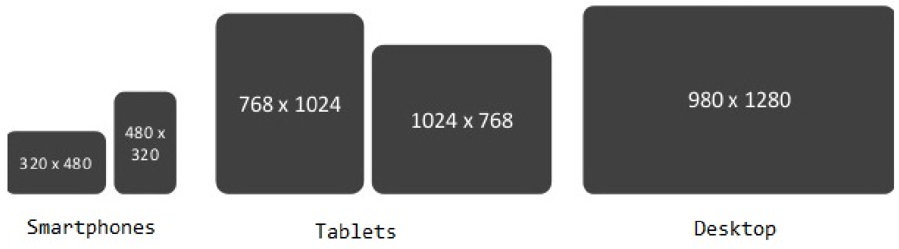


Figure 2.3: Different Viewports

2.2.2 Detection of MFP

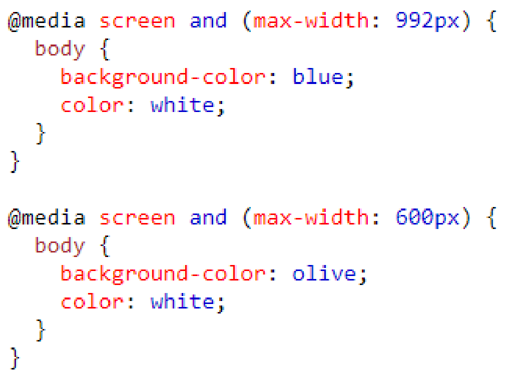
There are various manners by which a site can be improved to be more mobile friendly. In the beginning of web browsing in mobile, a typical methodology was to just develop another site which will only be rendered in mobile devices. Such sites were hosted at a different URL. When a user try to visit the website, it redirects the user to the mobile version of the website. The expense and time of building such a different site was high. To solve this issue, some commercial sites, like bMobilized [24] and Mobify [25], can consequently make a mobile-version site from the desktop version using some pre-developed layouts. However, a downside of these sites is that they can not hold the original design of the main desktop version of the website. In spite of the fact that having a different portable site could create mobile friendly issues, the maintenance cost of two different sites is quite high as they have to update the both sites simultaneously. This issue doubles the cost to of the organization to host their websites. Moreover, hosting a different site would not help improve search-rankings of the primary site, since the two sites are treated as different sites as they have different URLs. To get rid of from creating and maintaining two different sites, a software company may utilize responsive plan procedures. This sort of configuration utilizes CSS media inquiries to change the design of a page to the screen size ratio on which it will be shown dynamically. The benefit of this method over having different sites is that the URL of the site stays same as before. However, updating a current site into a completely responsive site is a time consuming task. Accordingly, fixing a current site might be a more financially beneficial plan than developing the site from the scratch. Besides, this responsive plan is probably going to ensure better user experience, it doesn't really block the chance of MFPs, since buggy styles might be incorporated. One methodology presents a novel procedure for dealing with MFPs by changing explicit CSS properties in the page and generating a fix. The fix utilizes CSS media inquiries to guarantee that the altered CSS is just utilized for mobile view only. It doesn't change the site when seen on a desktop. But the problem is with the symmetric structure. This approach does not hold the original symmetry of the website as it was before.

2.2.3 Search Based Technique

Search-based procedures investigate solution space effectively to an ideal solution. Search-based strategies start by choosing a sample set from the solution space. A fitness function evaluates their quality and gives a score to each candidate solution showing how good they perform [26]. This is called the target score or fitness score, helps in setting up a optimal search. In this process, if an improvement in the fitness score is seen over the first score, it demonstrates that the method is gradually approaching to the right way [27]. The algorithm would then be able to choose new candidates that have more possibility to accomplish further improvement of fitness score. This process continues until the best candidate solution is found which meets all the criteria.

2.2.4 CSS Media Query

A media query are CSS technology, consists of a syntax and at least one expression that limits the style sheets' scope by using media features, such as width, height, and color. Media queries allow CSS to only be applied when specific conditions are



```
@media screen and (max-width: 992px) {
  body {
    background-color: blue;
    color: white;
  }
}

@media screen and (max-width: 600px) {
  body {
    background-color: olive;
    color: white;
  }
}
```

Figure 2.4: CSS Media Query Code Snippet

met. Media queries, added in CSS3, helps in presentation of content be fitted to a specific range of output devices without having to change the content itself. Even more complex media queries can be composed using logical operators, including not, and, and only. The and operator is used for combining multiple media features together into a single media query, requiring each combined feature to return true in order for the query to be executed. The not operator is used to negate an entire media query. The only operator is used to apply a style only if the entire query matches, useful for preventing older browsers from applying selected styles. While using not or only operators, we must specify an explicit media type. We can combine multiple media queries in a comma-separated list; if any of the media queries in the list is true, the entire media statement returns true [28]. This is equivalent to or operator. We could write a media query that will only execute CSS, if the browser reaches a specific width. That means when a design is too large or too small, a media query can be used to detect the site width and serve CSS that appropriately rearranges the site's content [29].There is a code snippet for CSS media query in figure 2.4. Different segments of code is responsible for different viewports.

2.2.5 Evaluation Metrics for MFP

There are two important metrics available for the evaluation of the solution mobile friendly problems which are given below.

(a) Mobile Friendly Score: Goggle mobile friendly test gives a score between 1 to 100 on the basis of some properties of the target web page . Minimum 80 is the score to be passed as a mobile friendly site. The evaluation properties are:

   (i) Viewport Configuration: Triggered when Google detects your page does not specify a viewport, or specifies a viewport that does not adapt to different devices

   (ii) Font Legibility: Triggered when Google detects that text in the page is too small to be legible

   (iii) Use of Incompatible Plugins: Triggered when Google detects the use of plugins on your page

   (iv) Content to Viewport: Triggered when Google detects that the page content does not fit horizontally within the specified viewport size, thus forcing the user to pan horizontally to view all the content

   (v) Size and Proximity of Links: Triggered when Google detects that certain tap targets (e.g., buttons, links, or form fields) may be too small or too close together for a user to easily tap on a touchscreen

(b) Layout Distortion: Distortion is the alteration of the original shape (or other characteristic) of something. In web design, layout distortion means the faulty layout position of web elements. It can occur when an element overlaps on another one or more web elements. This score can be positive or negative because if the elements distortion creates a gap which is larger than the original gap, the length of that extra gap is counted as positive layout distortion. Besides, if the elements overlaps on one another, the length of overlapped content is counted as negative layout distortion.

2.3 Symmetric Structure

Symmetry is the quality of being made up of exactly similar parts facing each other or around an axis [30]. Symmetrical design relies on principles of balance, rhythm, proportion and unity to create a style with mirroring sides. Symmetry happens when the composition of design is distributed evenly around a central point or axis. This symmetry can be horizontal, vertical or radial in form. Symmetry creates balance, and balance in design creates harmony, order, and aesthetically pleasing results. It is found everywhere in nature, and is probably why we find it to be beautiful. Mathematical symmetry may be observed with respect to the passage of time; as a spatial relationship; through geometric transformations; through other kinds of functional transformations; and as an aspect of abstract objects [31]. Many structures, because of aesthetic and/or functional considerations, are arranged in symmetric patterns. Recognition of such symmetry will be identified and the use of this symmetry will be used to reduce the computational effort in analyzing such a structure.

In the context of web design, symmetric structure important. Symmetry in website means the equal distribution between the web elements on the basis of a symmetric axis. The structure depends on the composition of the objects. Moreover, it also depends on the ratio of height, width, padding of the HTML elements. Figure 2.5 shows a few example of symmetric layout design of web pages. We can see that, the segments of the web pages are equally distributed.

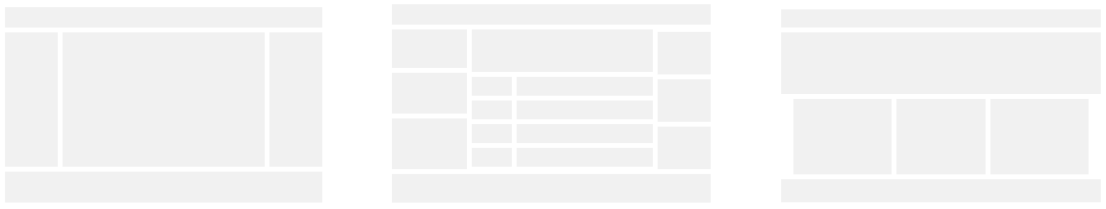


Figure 2.5: Examples of Symmetric Layout

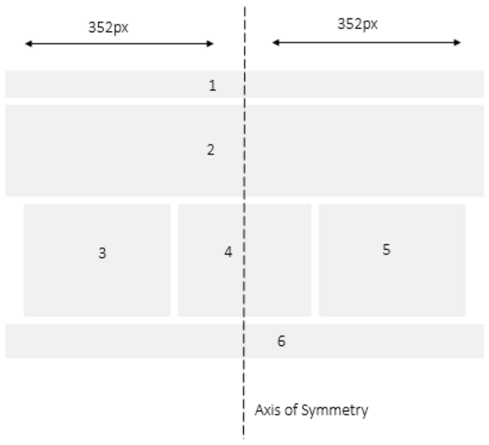


Figure 2.6: Axis of Symmetry

2.3.1 Symmetry Axis

[32] The importance of symmetric shapes has long been realized in the vision community. Such symmetries, when detected, contain useful information about the shape which can be used for shape description and for breaking shapes up into symmetric parts. In addition, it is clear that human observers are highly sensitive to symmetry and make use of it in their processing of images [33]. Symmetry axis or axis of symmetry is a straight line for which every point on a given curve has corresponding to it another point such that the line connecting the two points is bisected by the given line. It is a line that divides an object into two equal halves, thereby creating a mirror-like reflection of either side of the object. The word symmetry implies balance. Symmetry can be applied to various contexts and situations. Symmetry is a key concept in geometry which cuts the figure into two halves that are exact.

In the context of web design, axis of symmetry means the vertical axis along with the display viewport of the device which is used to render the web page. This axis determines the mirror image or the web page. Figure 2.6 shows the axis of symmetry of a given web page layout. The device used in the example is 704px. The web page has six segments.The axis divides the viewport in equal two mirror side. Now it is possible to calculate the exact distance of all of the segments and elements from this axis.

2.3.2 Types of Symmetry

There are three types of symmetry: reflection (bilateral), rotational (radial), and translational symmetry [34]. Each can be used in design to create strong points of interest and visual stability.

1. Reflection Symmetry: Reflection symmetry is also known as bilateral symmetry. It is the “mirror” effect, or when one object is reflected across a plane to create another instance of itself. The most common type of reflection we think of, and the most common we see in nature, is horizontal reflection, with the central axis being vertical. Reflection symmetry can take on any direction: vertical, diagonal, and anything in between.

2. Rotational Symmetry: Rotational symmetry (or radial symmetry) is when an object is rotated in a certain direction around a point.. In art and design, rotational symmetry can be used to portray motion or speed. Even on a static medium, rotational symmetry can convey action.

3. Translational Symmetry: Translational symmetry is when an object is relocated to another position while maintaining its general or exact orientation. These intervals do not have to be equal in order to maintain translational symmetry; they just need to be proportional. Translational symmetry can be used to create patterns, such as in the case of tiled website backgrounds

and repeating design elements. It can also be used strategically and more profoundly to create the feeling of motion and speed just like rotational symmetry.

In the context of web design, there are more type symmetry like horizontal symmetry, vertical symmetry, radial symmetry and color symmetry. Approximate horizontal symmetry is the most common form of symmetrical design. It is based on the principle of horizontal symmetry but items do not have to half perfectly. Sides of a form may be slightly different in shape or size. Radial symmetry occurs when an object is symmetrical around a center point. Radial symmetry can create the most distinct sense of harmony but can be hard to replicate in many design arenas. The shape of the screen of computer, tablet or phone is likely rectangular, eliminating the possibility of creating a perfectly radial design. Almost all designs using a radial symmetry technique are circular or square in form. Vertical symmetry is the counterpart to horizontal symmetry. Rather than basing the center point from top to bottom, it is created from left to right; if the top and bottom halves mirror, the design is vertically symmetrical. Vertical symmetry is most commonly used when grouping rows of similar elements. My solution approach is concerned with approximate horizontal symmetry and vertical symmetry.

2.3.3 Evaluation Metrics for Symmetry

There are some metrics to evaluate symmetry of a web page. To measure the equilibrium along a vertical axis, balance score and proportion score are useful metrics. These metric are calculated based on the approximate horizontal symmetry and vertical symmetry of the web page. In the context of web design, these two metrics are mentioned below.

(a) Balance Score: It is a metric for measuring equilibrium along a vertical axis in the layout. Balance is the equal distribution of visual weight in a design.

Visual balance occurs around a vertical axis; our eyes require the visual weight to be equal on the two sides of the axis. We are bilateral creatures and our sense of balance is innate. When elements are not balanced around a vertical axis, the effect is disturbing and makes us uncomfortable. Balance score is calculated by the summation of symmetrical balance, vertical balance and horizontal balance. The properties of balance are:

   (i) The number of elements in each segments on both sides.
   (ii) The headline and sub-headline are equally long on both sides or not.
   (iii) The call to action button is centered or not.
   (iv) The number of elements under the heading on both sides

(b) Proportion Score: It is the ratio between the dimensions of elements. The ratio is calculated by dividing the height of any element by its length. The proportion of every elements in a web page needs to be similar to maintain symmetric structure. The proportion score of a web page is calculated by the average of individual proportion score of each elements of that web page.

2.4 Summary

A brief description of web fundamentals, mobile friendly problems is discussed in this chapter. It also deals with how and why mobile friendly problems occur in a web page. The affects of mobile friendly problem and symmetric structure are also discussed. The evaluation metrics for symmetric structure of web application are also presented in this chapter. Besides, some aspects of the solution of mobile friendly problems are also discussed. The next chapter exhibits the relevant existing works that have been performed in the literature.

Chapter 3

Literature Review of Improving Symmetric Structure during Resolution of MFP

In this chapter, the existing mobile friendly problem solving approaches will be discussed. In the literature, most of them have proposed the detection and resolving process of presentation failure of web applications [35, 36, 37, 38, 12]. A few have proposed the resolution process of mobile frienldy problems in web pages [13]. Some of them have shown the significance of visual symmetry as an important factor in our aesthetic perception of websites [7, 8, 9]. Based on problem domain, four types of knowledge area are visible from the literature which are listed below.

1. Layout Faults
2. Presentation Failures
3. Cross Browser Issues
4. Mobile Friendly Problems

It has been observed from the literature that the first three are mostly concerned with the presentation issues of responsive web design. The last one are concerned with mobile friendly problem. In the remaining sections, all these approaches are discussed.

3.1 Layout Faults

Since software needs modification very frequently, the developer may struggle to recall the initial CSS modification that cause layout fault problem. Therefore, this requires more effort to diagnose and fix it. Each time a change is made, all resolutions need to be manually rechecked for unintentional side-effects. This is a time-consuming and error-prone process – in viewing a large number of resolutions and layouts, developers and testers can easily miss faults that ultimately make their way into the live website. There is an automated method for detecting potential faults in the layout of responsively designed pages [35].

This approach models the layout of a responsive web page across a series of viewport widths, explicitly taking account of the key aspects of responsive layout, namely the changing visibility, width, and relative alignment of web page elements. They refer to this model as the Responsive Layout Graph (RLG). To generate the RLG, the approach first inspects the layout of the web site in question at a series of viewport widths, gaining an overall understanding of the responsive layout and recording its DOM at each viewport width. This represents the starting point for the RLG construction algorithm. Next, the extraction of alignment constraints begins by creating an element hierarchy at each viewport width where an element's parent is the smallest element to completely contain it. Finally, the method examines the width of each element and its parent element across all viewport widths, thereby producing a set of width constraints for that particular element [35].

To experimentally evaluate this approach, they implemented a tool, called REDECHECK, and applied it to 5 real-world web sites that vary in both their approach to responsive design and their complexity. The experiments reveal that REDECHECK finds 91% of the inserted layout faults [35].

This paper focused on layout faults. This approach can not detect mobile friendly problems. Moreover, new HTML elements namely header, footer, section etc have been introduced recently. This approach does not work properly with these elements [35].

3.2 Presentation Failures

Web applications ensure organizations to publish their products and services around the globe. It is tough for software developers to design their websites to accommodate the expansion and contraction of text after it is translated to another language. There is an automated technique for detecting when a web page's interface has been distorted due to internationalization efforts and identifying the HTML elements or text responsible for the mentioned problem [38].

In the first phase of the approach, it analyzes the page under test and original page to build an layout graph of each.The layout graph (LG) is a model of the visual relationships of the elements of a web page. After computing the nodes of the graph, the second step of the approach is to define a function, which annotates each edge in the graph with a set of visual relationships. In the second phase, the approach compares the two LGs produced by the first phase in order to identify differences between them. In the final step, the method analyzes the set of tuples, identified in the second phase and generates a list which contains HTML elements by their ranks and text that may be the cause for the mentioned presentation failures.

They implemented this approach as a Java prototype tool, GWALI (Global

Web Applications' Layout Inspector). Their subject pool contained 54 different web applications. Their results of the evaluation show that this approach can detect IPFs and the precision is 91% precision and the recall is 100% [38].

This approach works for presentation failure issue of web application. This solution is too specific to generalize to the broader domain.

3.3 Cross Browser Issues

The continually expanding number of internet browsers with which clients can get to a site has presented new difficulties in controlling visual issues. Contrasts in how different programs render HTML and CSS rules can create Cross Browser Issues (XBIs) — differences in the appearance of a site across various web browsers [37]. Designers must check that their sites render and work reliably across different web browsers as expected under the circumstances. Moreover, the modern designs and styles of current web applications make it hard to recognize the UI components liable for the observed XBI. This paper proposes a strategy for consequently fixing format XBIs in sites utilizing search based algorithm.

This methodology produces fixes utilizing search based algorithm. The underlying part of the methodology includes acquiring the fix set of XBIs. After that, the algorithm finds the main cause that creates XMIs. The main step in this method extracts CSS properties corresponds to the XBI. It produces candidate fixes for each underlying causes. The fix is a combination of CSS properties that is selected by a improved fitness score. After that, the algorithm settles on a decision to the "searchForBestRepair" system to get the best fix. The last phase of the algorithm decides it terminates or continue to another cycle [37].

They developed this approach in a tool called "XFix" in Java. For evaluation they used 15 real-world subjects. The average of decrease in XBIs is 86% and the median is 93%. This proves that XFix was able to find XBI fixes. 78% of the client evaluations showed an improved likeness of the updated version, inferring that the consistency of the subject pages had improved with recommended fixes. 14% of the client ratings proved the improved consistency, and just 8% of the client evaluations revealed a negative consistency. XFix had a median runtime of 14 minutes to resolve XBIs [37].

The XFix technique repairs layout Cross Browser Issues (XBIs) — presentation failures. In this domain, the "correct" presentation of the page is available through one of the browsers, the layout of which must be mimicked in another. Mobile friendly problems entail a different approach, in which the presentation of a page must be changed to correct a series of identified issues. There is no correct reference rendering available to the repair process, which must also maintain as much of the original aesthetics of the page as possible [37].

3.4 Mobile Friendly Problems

Designing websites to be mobile friendly ensures that the pages of that website perform well on all devices. The desktop version of a site might be difficult to view and use on a mobile . The user has to pinch or zoom in order to read the content from the version that's not mobile-friendly. There is a novel automated approach for solving mobile friendly problems in web application [13].

They proposed a way to dynamically create CSS patches that can improve the mobile friendliness of a site [13]. To effectively get the best fix, this methodology checks different aspects of the problem to evaluate metrics identified by layout distortion and the calculation of the solution. This technique will solve three specific mobile friendly problem such as:

1. Content sizing: This problem occurs when a web page overflows the width of the viewport of a device. The user needs zoom out or scroll horizontally to view the content.

2. Font sizing: When a web page is rendered in a mobile device, the font sizes of the page might not be resized with the viewport of the device. For this, the user faces trouble to view the content properly because of the small font size.

3. Tap target spacing: When a user views a site on a mobile device, the target elements which are needed to tap on to do an action may be too small to tap. They need to zoom into the site to tap the targets.

The approach is divided into 3 phases[13]:

1. Segmentation: This stage examines the structure of the page to distinguish portions - sets of HTML components whose properties ought to be changed together to keep up the visual consistency of the fixed page.

2. Localization: The subsequent stage recognizes the parts of the website that need to be isolated to find its mobile friendly problems. These problems are detected by Mobile Friendly Oracle (MFO).

3. Repair: This phase computes a repair CSS set for the web page. It identifies a patch that improves the web page's mobile friendliness score. It also ensures that the generated patch does not significantly distorts the layout of the web page.

They analyzed 38 real-world subjects collected from the best 50 most visited websites over all seventeen categories reported by Alexa. The results for effectiveness were that 95% (36 out of 38) of the subjects passed the GMFT after Applying MFix's proposed CSS repair fix. On average, MFix moved forward the MF score of a subject by 33%. The entire running time (RQ2) required by this approach for the various subjects extended from 2 minutes to 10 minutes, averaging a small less than 5 minutes. In terms of the measuring metric, the repaired version was evaluated to have improved visual value as it were 38% of the time [13].

This approach solves mobile friendly problems of a website but it can not maintain the original symmetric structure. Therefore, the repaired version has a lower attractiveness. According to their evaluation, the repaired version had a lower attractiveness 62% of the time[13].

Chapter 4

Symmetric Structure during Resolution of Mobile Friendly Problems in Web pages

As discussed in the previous chapter, mobile friendly problems should not be solved without maintaining the symmetric structure of the web application. MFP solution techniques in the literature use search based technique to find the optimal solution. But these solutions do not focus on the structure of the web page. As a result, both MFP solution and symmetric structure is needed to get a combined output. These problem has been also fit into the search based solution technique. To support the fitness function of this method, two more metrics named balance score and proportion score are introduced. The details of these metrics is described in the 2nd chapter.

4.1 Overview of the Approach

This section contains the core parts of the technique which are later described in details throughout this chapter. The following points are the main constitutes of the proposed approach.

The approach will find the faulty HTML elements that cause asymmetry. Moreover, the CSS attributes and the values of the elements will be identified. For example, an image containing absolute positioning, fixed height and fixed length can cause asymmetry. To detect the asymmetry, the approach will check the balance between the two sides of a web page according to the centralized vertical axis. The approach needs to identify all these problematic properties and values that may cause asymmetry.

After that, patches will be generated that will modify some CSS properties like position, height, width, padding etc. so that the whole page will be symmetrically structured. However, improving symmetry can cause layout distortion. Layout distortion means overlapping of the segments of a website. Therefore, the approach will generate patches that will maintain trade-off between symmetry and layout distortion. For the afore-mentioned example, the approach will update the position property value (static, relative or sticky) to achieve the most symmetric structure possible for that web page. However, the updated element can overlap on other elements that cause segment over-lapping. To resolve this, the approach will reposition the overlapped segments maintaining symmetry. The overview of the solution approach is shown in figure 4.1.

4.1.1 Segmentation of the web page

The primary stage analyzes the structure of the page to recognize fragments - sets of HTML components whose properties should be balanced together to preserve the visual consistency of the repaired page. An illustration of a segment may be a arrangement of text-based links in a navigation bar where on the off chance that the text style measure of any interface within the section is little, at that point all of the nav links ought to be balanced by the same amount to preserve the links' visual consistency. To maintain consistency, the fix value of a target element has

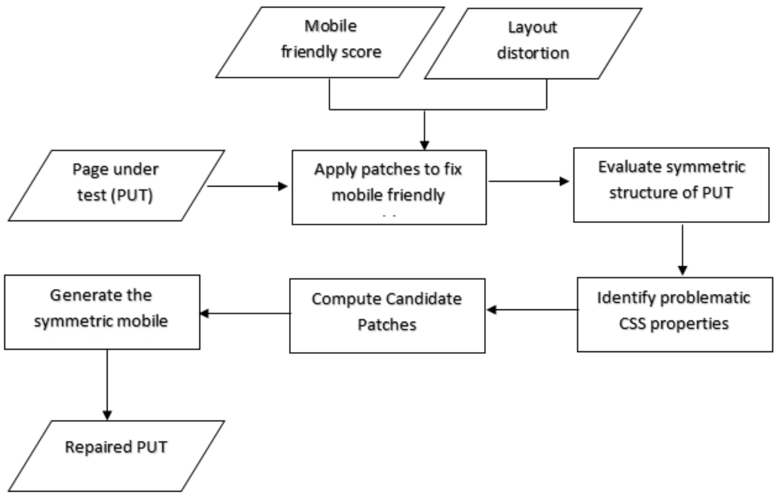

Mobile friendly score
Layout distortion
Page under test (PUT)
Apply patches to fix mobile friendly
Evaluate symmetric structure of PUT
Generate the symmetric mobile
Compute Candidate Patches
Identify problematic CSS properties
Repaired PUT

Figure 4.1: Overview of the solution approach

to be close to the related elements. That why, segmentation is needed.

The document object model (DOM) tree of the target web page needs to parsed to identify the desired segments from a web page. There are several types of segmentation process, such as Block-o-matic [39], clustering based partitioning [40], correlation clustering[41]. I have chosen to use visual representation based segmentation process called VIPS [42]. In the VIPS algorithm, the vision-based content structure of a page is deduced by combining the DOM structure and the visual cues. First, DOM structure and visual information, such as position, background color, font size, font weight, etc., are obtained from a web browser. Then, from the root node, the visual block extraction process is started to extract visual blocks of the current level from the DOM tree based on visual cues. Every DOM node is checked to judge whether it forms a single block or not. If not, its children will be processed in the same way. When all blocks of the current level are extracted, they are put into a pool. Visual separators among these blocks are identified and the weight of a separator is set based on properties of its neighboring blocks. After constructing the layout hierarchy of the current level, each newly produced visual blocks is checked to see whether or not it meets the granularity requirement. If no, this block will be further partitioned. After all blocks are

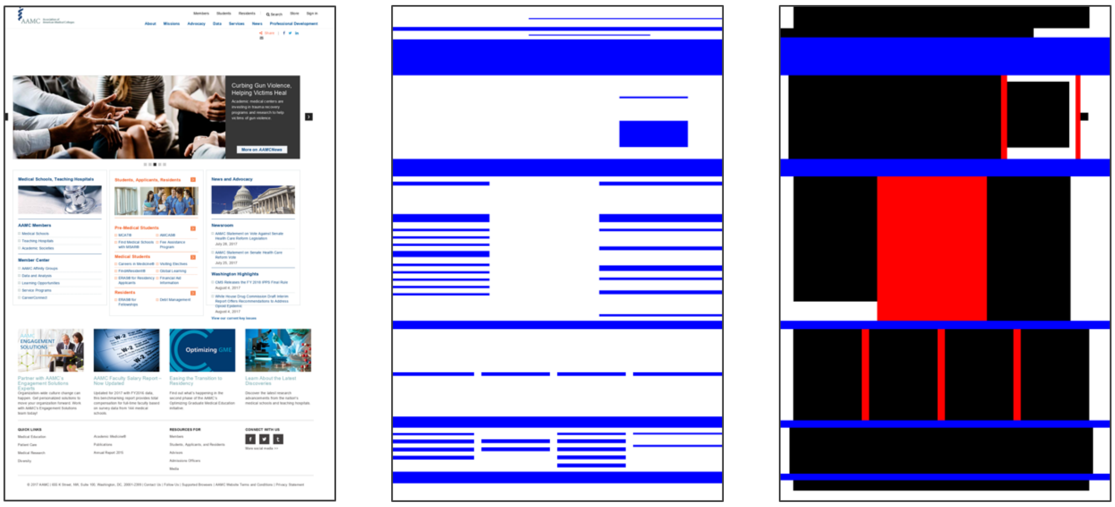


Figure 4.2: Example of Web Page Segmentation

processed, the final vision-based content structure for the web page is outputted. Below we introduce the visual block extraction, separator detection and content structure construction phases respectively. The steps of VIPS is shown in figure 4.2. We can see that, the segments are the header content, multiple content pane and the footer.

4.1.2 Applying Mobile Friendly Fix

After segmentation, this approach incorporates the existing mobile friendly fix[13]. The overview of the process is given below:

1. Identifying Problematic Segments: To detect mobile friendly problems in a web page, google mobile friendly test (GMFT) is used. The GMFT tool identifies the problems with the faulty elements and their CSS properties. Then it is easy to determine which segments have faulty elements by analyzing the CSS properties. This approach applies a update to the segments to recognize which ones may be tricky with regard to that mobile friendly issue. The issue with the CSS properties is, they inherit styles from their parent HTML elements. So this approach introduce Property Dependence Graph (PDG). This approach defines three type of PDG: font, content size

and tap target PDG.

2. Repair: The repair step uses two metrics named mobile friendliness and layout distortion. The Google mobile friendliness test scores each web page on the basis of mobile friendliness. This score has a range of 0 to 100. The measure of update in a layout can be controlled by building models that express the relative visual positioning among and inside the parts of a page. To decide the edge marks, the methodology registers the Minimum Bounding Rectangles (MBRs) of each section [43]. This is done by finding the largest and smallest X and Y coordinate values of the components, which can be obtained by analyzing the DOM of the page.

3. Computing Candidate Patches: The best CSS patch balances between the mobile friendliness and layout distortion.It irritates adjustment factors as to find the appropriate fix that vary a little from the original web page. This approach finds that the mean and standard deviation values are the best in context of Gaussian distribution. Finally, this approach included an amendment factor to permit the way to get the optimal solution.

4. Generating Final Patch: The method creats a set of fixes. The CSS properties are updated according to the fix result for all nodes in the PDG. A CSS media query is formed by these updated CSS style values. When an user access the page, this media query cause it to be rendered.

4.1.3 Detecting Faulty Elements

In this stage, the approach finds the faulty HTML elements that cause asymmetry. Moreover, the CSS properties and the values of the target elements will be identified. For example, an image containing absolute positioning, fixed height and fixed length can cause asymmetry. To detect the asymmetry, the approach will check the balance between the two sides of a web page according to the centralized vertical axis. The method has to identify all these problematic properties and values that may cause asymmetry.

The CSS properties inherit values from their parent. For this issue, this approach introduces Property Dependence Graph (PDG). PDG is a directed graph which has a form (N, E, T) where N is a node that represents an HTML elements in the graph, E is a set of directed edges. These edges holds a dependency relationship between the corresponding elements. T is a function that maps each edge to a set of tuples. That is how the inherited CSS properties is achieved for every HTML elements of the document object model.

After getting all the HTML elements along with the CSS properties, the approach detects the faulty elements that cause asymmetry to the web page. This detection is done separately for each segments of the web page. The symmetric evaluation of the HTML elements is concluded by two metrics. They are:

(a) Balance Score: It is a metric for measuring equilibrium along a vertical axis in the layout. Balance is the equal distribution of visual weight in a design. Visual balance occurs around a vertical axis. Our eyes require the visual weight to be equal on the two sides of the axis.Balance score is calculated by the summation of symmetrical balance, vertical balance and horizontal balance. The balance score of an individual element, $B_i = (N_i, H_i, C_i, E_i)$ is defined as follows:

   (i) N_i The number of elements in each segments on both sides.

   (ii) H_i The distance from the closest headline and sub-headline on both sides or not.

   (iii) C_i The distance from the closest call to action button according to the center.

   (iv) E_i The number of elements under the heading on both sides

The total balance score is the summation of each balance score of the targeted web page. If the the total balance score for a page p is $B(p)$ and total number of elements in that page is n, then

$$B(p) = \frac{(n * 100) - \sum_{i=1}^{n}(N_i, H_i, C_i, E_i)}{n * 100}$$

(b) Proportion Score: It is the ratio between the dimensions of elements. The ratio is calculated by dividing the height of any element by its length. The proportion of every elements in a web page needs to be similar to maintain symmetric structure. The proportion score of a web page is calculated by the difference of individual mean proportion score of each mirror side of that web page. The solution is looking for a proportion score close to zero. If the height of an HTML element e is H_e and width is W_e, the proportion score of that element is P_e.

$$P_e = \frac{H_e}{W_e}$$

The web page needs to divided into two sides on the basis of vertical axis. Then the elements will be separated into two sets such as left set and right set. The elements, which are overlapped on the vertical axis, are the common members of the two sets. If the proportion score of left set is $P_L s$ and right set is $P_R s$,

$$P_L s = \frac{\sum_{i=1}^{n_l} P_e}{n_L}$$

where the number of elements in left side is n_L and

$$P_R s = \frac{\sum_{i=1}^{n_R} P_e}{n_R}$$

where the number of elements in left side is n_R. Therefore, the final proportion score of a page W is P_W,

$$P_W = P_Ls - P_Rs$$

These two metrics are used to detect the faulty elements in that web site which cause the distortion of symmetry. Moreover, the approach gets the faulty segments by analysing the faulty HTML element's segment.

4.1.4 Generate Symmetric Mobile Friendly Patch

The idea behind the solution is to achieve optimal solution efficiently. Search based technique has been used to generate candidate patches to get symmetric mobile friendly web page. The value of balance score and proportion score has been changed dynamically to gain highest balance score (close to 1) and lowest proportion score (close to 0). Search-based strategies start by choosing a sample set from the solution space. For this, the method iterates over each of the elements in the site. For every component, it repeats over every node of the PDG, beginning with the root node, and updates value that will be referred to CSS property. When new property estimations have been completed for all nodes in the PDG, the method creates a bunch of fixes. The values have been updated according to achieve the highest balance score and lowest proportion score.

New values are made into CSS style declarations by converting it into a CSS selector. A CSS media query is formed by these updated CSS style values. When an user access the page in a mobile viewport, this media query cause it to be rendered with the updated CSS patch. Figure 4.3 shows the whole process of generating candidate fixes and how the symmetric mobile friendly patch is generated using balance score and proportion score.

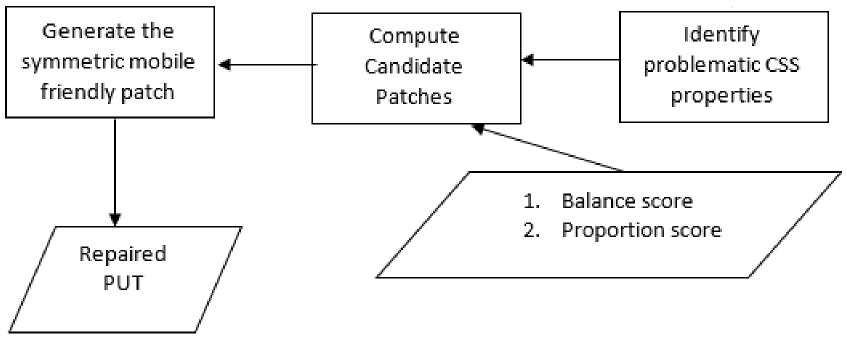

Generate the symmetric mobile friendly patch
Compute Candidate Patches
Identify problematic CSS properties
1. Balance score
2. Proportion score
Repaired PUT

Figure 4.3: Generate Symmetric Mobile Friendly Patch

4.2 Summary

A search based technique has been proposed that uses dynamic balance score and proportion score to address an optimal solution to get symmetric mobile friendly CSS patch. The solution can be applied to any types of viewports. Because the size parameters are configurable in the media query according to the accessed device viewport size.

Chapter 5

Implementation and Result Analysis

This chapter contains the implementation details, experiments, result discussion of the proposed approach. The discussion of the results to explain some important insights are also presented. This chapter also describes the subjects, tools that is used in the implementation, environment setup and evaluation questions. To evaluate the approach a user based study is also been conducted which is described in the later part of this chapter.

5.1 Evaluation Questions

To evaluate the approach for generating symmetric mobile friendly patch, I designed some evaluation questions to measure its effectiveness of its solutions. The research questions which were considered in this research:

SQ1: How to identify CSS properties and values of the HTML elements that cause asymmetry of a web page?

SQ2: How to generate patches that improve symmetric structure maintaining trade-off between symmetry and layout distortion in different viewports?

5.2 Implementation Details

The proposed methodology was implemented in Java using Eclipse Photon 4.8 Integrated Development Environment (IDE) [44]. Moreover, I used python 3 using virtual studio code to develop the analyzer tool named UIAnalyzer. An Object Oriented Programming language such as Java provides extensive support for polymorphism, inheritance and encapsulation which are necessary for reusability. This is why it was used to implements the approach. The following tools and libraries were used to develop the system.

- GMFT: I have used Google mobile friendly test api to indentify the mobile friendly problems in a web application.Mobile Friendly Test is a tool that allows to easily carry out a mobile site test telling about a website's score in terms of mobile responsiveness.

- Chrome Emulator: To render the document object model of a webpage and to get information about XPath of the HTML elements, an emulated Chrome browser v65.5 has been used for mobile experience.

- Selenium WebDriver: Selenium is a portable software-testing framework for web applications. Selenium WebDriver accepts commands and sends them to a browser. This is implemented through a browser-specific browser driver, which sends commands to a browser and retrieves results. Most browser drivers actually launch and access a browser application (such as Firefox, Chrome, Internet Explorer, Safari, or Microsoft Edge). Selenium Webderiver is used to get all the HTML elements parsed.

- jStyleParser: jStyleParser is a CSS parser written in Java. It has its own application interface that is designed to allow an efficient CSS processing in Java and mapping the values to the Java data types. It parses CSS 2.1 style sheets into structures that can be efficiently assigned to DOM elements [45]. I have used this parser for detecting defined CSS properties for every HTML nodes in a web page.

- BeautifulSoup: Beautiful Soup is a Python package for parsing HTML and XML documents (including having malformed markup, i.e. non-closed tags, so named after tag soup). It creates a parse tree for parsed pages that can be used to extract data from HTML, which is useful for web scraping. I used beautifulsoup 4.6.3 to develop the UIAnalyzer tool.

The implementation was executed on an operating system with following configurations:

- Operating System: Windows 10 Enterprise
- RAM: 12 GB
- CPU: 2.50GHz Intel Core-i5 Processor
- Platform: 64 bit

5.3 Subjects

For result analysis, 54 websites were selected from the most popular websites list across all categories listed by MOZ top 500 websites [46]. The subjects are listed in table 5.1 and 5.2.This table also shows the popularity rank and total number of HTML elements of each subject website. The number of elements can indicate an approximation for the complexity and size of the subject.

I selected MOZ website list as the source because it gives the popularity based ranking and a mix of different layouts. I have used HTTrack [47] to download all the HTML files, images, CSS and javascript files from the website. Then I removed the advertisements part to use these subjects in offlie mode also.

Table 5.1: Subjects used in evaluation-1

ID	**URL**	**Rank**	**#HTML**
1	aamc.org	23	598
2	arxiv.org	21	381
3	us.battle.net	2	615
4	bitcointalk.org	25	1302
5	blizzard.com	33	313
6	boardgamegeek.com	31	4474
7	bulbagarden.net	26	151
8	coinmarketcap.com	8	1964
9	correios.com.br/para-voce	14	769
10	dict.cc	20	633
11	discogs.com	26	5738
12	drudgereport.com	23	779
13	finalfantasyxiv.com	37	61
14	flashscore.com	16	6621
15	fragrantica.com	35	1091
16	forum.gsmhosting.com/vbb	39	2618
17	intellicast.com	38	1393
18	irctc.co.in	34	1031
19	irs.gov	14	569
20	leo.org	31	990
21	letour.fr	3	1260
22	lolcounter.com	30	1257
23	mmo-champion.com	29	1903
24	myway.com	42	135
25	ncbi.nlm.nih.gov	2	833
26	nexusmods.com	28	2108
27	nvidia.com	20	719
28	rotoworld.com	41	2523
29	sigmaaldrich.com	37	141
30	us.soccerway.com	30	2708

Table 5.2: Subjects used in evaluation-2

ID	**URL**	**Rank**	**#HTML**
31	square-enix.com	30	198
32	travel.state.gov	26	440
33	weather.gov	18	1101
34	bom.gov.au	48	685
35	wiley.com	14	460
36	onlinelibrary.wiley.com	33	824
37	wowprogress.com	46	2828
38	xkcd.com	48	121
39	telegraph.co.uk	115	590
40	washingtonpost.com	116	2145
41	ft.com	117	1245
42	hatena.ne.jp	118	578
43	networkadvertising.org	123	963
44	businessinsider.com	127	998
45	steampowered.com	128	950
46	un.org	130	1645
47	plesk.com	133	580
48	aliexpress.com	138	254
49	terra.com.br	139	740
50	bit.ly	143	650
51	deezer.com	214	189
52	thetimes.co.uk	217	1250
53	mashable.com	220	1444
54	cnbc.com	221	254

5.4 User based case study

I conducted a user based case study by doing a survey. The purpose of this survey is to evaluate the visual appeal of the updated web page that is incorporated with symmetric mobile friendly patch. The participant were asked to compare the previous and updated versions of the subjects in that survey. For each subject, the survey included a screenshot of the original and updated pages when viewed in a viewport of the mobile device. The questions of the survey were:

1. Which version between two screenshots (One is mobile friendly, other one is symmetric mobile friendly) is more visually attractive?

2. Which version they would prefer more to use in a mobile device?

3. Rate the each version of the target page on a scale of 10.

I used Google Form service to do this survey. For the survey, I had 25 participants who are currently working as software developers in different multi-national software companies. Based on the result of the survey, the participants preferred using symmetric version 48 out of 54 subjects. Only 6 subjects had a negative preference for using the symmetric version. On average, the rating of updated symmetric mobile friendly version had a 21% improvement than the mobile friendly version. I used Wilcoxon signed-rank test[48] to confirm the result as significant because the calculated $p - value = 2.68 \times 10 - 13 < 0.05$

5.5 Result Analysis

The effectiveness of the approach is dependent on two metrics such as balance score and proportion score of a web page. These range of two metrics is 0 to 1. If the balance score is close to 1 and the proportion score is close to 0, it is evaluated as more effective. First I have calculated these metrics for the version with mobile friendly patch. Then I have calculated these metrics for the updated version of the website with symmetric mobile friendly patch. As we are targeting to increase the balance score and decrease the proportion score, it is easy to evaluate the effectiveness of the approach by analyzing the difference of the two versions of the subjects.

In the table 5.3 and 5.4, a indicates the value of mobile friendly version and b indicates the value of symmetric mobile friendly version. Here, B represents Balance score, P represents proportion score, G represents gain and L represents loss. we can see that, 49 out of 54 subjects have gained balance score. Therefore the effectiveness of the approach on the basis of balance is 90.7%. Moreover, 52 out of 54 subjects have lost proportion value. The effectiveness on the basis of proportion score is 96.2%. The figures 5.1 and 5.2 show the distribution of balance

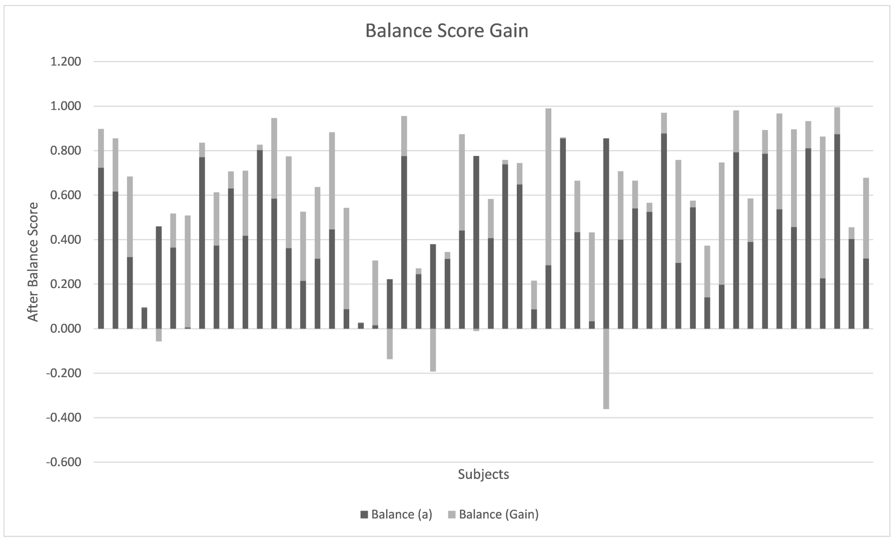


Figure 5.1: Distribution of Balance Score Gain

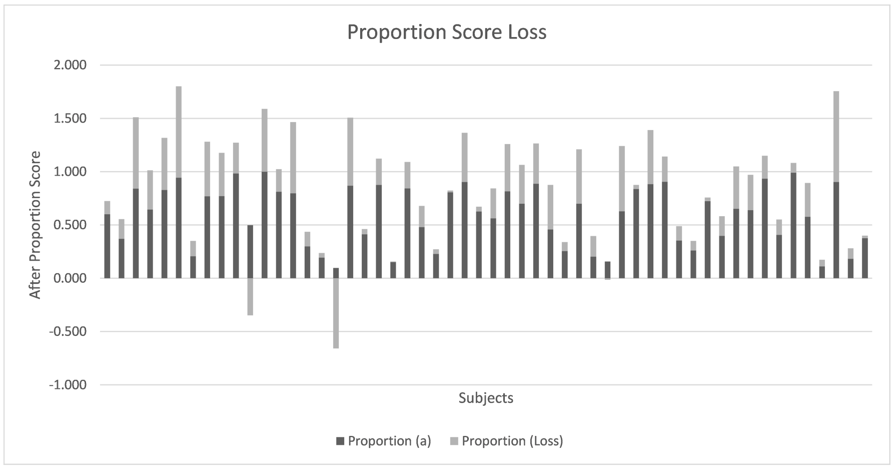


Figure 5.2: Distribution of Proportion Score Loss

score and proportion score. It shows that the symmetric mobile friendly version generates better patch than the previous patch in terms of symmetry.

Table 5.3: Evaluation of Balance Score and Proportion Score

ID	URL	B (a)	B(b)	B(G)	P (a)	P (b)	P(L)
1	aamc.org	0.723	0.897	0.174	0.601	0.477	0.124
2	arxiv.org	0.616	0.856	0.240	0.371	0.188	0.184
3	us.battle.net	0.322	0.684	0.362	0.842	0.173	0.669
4	bitcointalk.org	0.093	0.098	0.005	0.645	0.276	0.369
5	blizzard.com	0.460	0.402	-0.058	0.828	0.338	0.490
6	boardgamegeek.com	0.365	0.518	0.153	0.943	0.086	0.858
7	bulbagarden.net	0.005	0.509	0.503	0.207	0.062	0.145
8	coinmarketcap.com	0.770	0.836	0.066	0.768	0.255	0.513
9	correios.com.br	0.374	0.613	0.239	0.770	0.363	0.407
10	dict.cc	0.630	0.707	0.076	0.983	0.694	0.290
11	discogs.com	0.417	0.710	0.293	0.500	0.848	-0.348
12	drudgereport.com	0.803	0.827	0.024	1.000	0.409	0.591
13	finalfantasyxiv.com	0.584	0.947	0.363	0.813	0.602	0.211
14	flashscore.com	0.362	0.775	0.413	0.798	0.130	0.668
15	fragrantica.com	0.215	0.526	0.311	0.299	0.163	0.137
16	forum.gsmhosting.com	0.315	0.637	0.322	0.194	0.151	0.043
17	intellicast.com	0.446	0.883	0.438	0.098	0.756	-0.658
18	irctc.co.in	0.088	0.543	0.455	0.869	0.232	0.638
19	irs.gov	0.025	0.029	0.004	0.413	0.365	0.048
20	leo.org	0.015	0.307	0.292	0.877	0.632	0.245
21	letour.fr	0.222	0.085	-0.137	0.153	0.147	0.005
22	lolcounter.com	0.775	0.956	0.180	0.844	0.597	0.247
23	mmo-champion.com	0.245	0.271	0.026	0.480	0.281	0.199
24	myway.com	0.380	0.187	-0.193	0.228	0.184	0.044
25	ncbi.nlm.nih.gov	0.314	0.345	0.031	0.808	0.792	0.016
26	nexusmods.com	0.441	0.874	0.433	0.904	0.443	0.461
27	nvidia.com	0.776	0.765	-0.011	0.626	0.580	0.046
28	rotoworld.com	0.407	0.583	0.176	0.563	0.282	0.281
29	sigmaaldrich.com	0.739	0.758	0.020	0.816	0.371	0.445
30	us.soccerway.com	0.648	0.745	0.096	0.700	0.336	0.364
31	square-enix.com	0.087	0.216	0.129	0.888	0.510	0.378
32	travel.state.gov	0.285	0.989	0.705	0.459	0.041	0.418

Table 5.4: Evaluation of Balance Score and Proportion Score-2

ID	**URL**	**B (a)**	**B(b)**	**B(G)**	**P (a)**	**P (b)**	**P(L)**
33	weather.gov	0.855	0.860	0.005	0.256	0.173	0.083
34	bom.gov.au	0.433	0.666	0.232	0.699	0.189	0.510
35	wiley.com	0.033	0.433	0.399	0.204	0.010	0.193
36	onlinelibrary.wiley.com	0.855	0.494	-0.361	0.158	0.172	-0.015
37	wowprogress.com	0.401	0.708	0.307	0.628	0.014	0.614
38	xkcd.com	0.540	0.665	0.125	0.839	0.801	0.038
39	telegraph.co.uk	0.525	0.566	0.041	0.884	0.376	0.507
40	washingtonpost.com	0.878	0.970	0.093	0.905	0.668	0.237
41	ft.com	0.295	0.758	0.463	0.355	0.220	0.135
42	hatena.ne.jp	0.545	0.576	0.030	0.262	0.174	0.088
43	networkadvertising.org	0.141	0.373	0.232	0.722	0.687	0.035
44	businessinsider.com	0.198	0.747	0.549	0.398	0.212	0.186
45	steampowered.com	0.792	0.981	0.188	0.653	0.256	0.396
46	un.org	0.389	0.585	0.196	0.640	0.309	0.331
47	plesk.com	0.786	0.893	0.107	0.935	0.720	0.215
48	aliexpress.com	0.537	0.967	0.430	0.408	0.266	0.142
49	terra.com.br	0.456	0.896	0.440	0.990	0.899	0.091
50	bit.ly	0.810	0.932	0.122	0.577	0.259	0.318
51	deezer.com	0.226	0.863	0.637	0.111	0.048	0.063
52	thetimes.co.uk	0.874	0.994	0.121	0.904	0.053	0.851
53	mashable.com	0.403	0.456	0.053	0.184	0.085	0.098
54	cnbc.com	0.315	0.678	0.363	0.376	0.352	0.024

5.6 Summary

This chapter shows the effectiveness of symmetric mobile friendly approach in two different ways. The first one is the metric method. The repaired version has achieved better score in terms of balance score and proportional score. Table 5.5 shows that the mean value of balance gain is 0.20 and the mean value of proportion loss is 0.25 which are greater that 0. It confirms that the repaired versions have performed better than the previous mobile friendly technique. A survey based evaluation also shows that 88% subjects, out of 54, have been considered as more preferable than the previous version by the users.

Table 5.5: Summary of Evaluation Metrics

	Max	**Min**	**Average**
Balance Gain	0.705	-0.361	0.202
Proportion Loss	0.858	-0.658	0.253

Chapter 6

Conclusion

This thesis proposes an automated repair technique that incorporate mobile friendly solutions and symmetric structure of web application. For this, a search based approach has been devised that considers coordination between mobile friendly problem and symmetry problem. This chapter discusses the summary of the thesis with respect to its contribution and achievements. The threats to validity of this work are presented next. Finally, it is concluded with some future research directions.

6.1 Thesis Summary

This thesis proposes a repair technique that generates symmetric mobile friendly patches for a web page. To determine symmetry, this approach introduces two symmetry based metrics named balance score and proportion score. Using the values of these metrics in a search based approach, this technique generate candidate fixes for the website. The best fix is generated by maximizing the balance score and minimizing the proportion score.

To evaluate the solution of symmetric mobile friendly approach, I used two different ways. The first one is the metric method. The repaired version has achieved better score in terms of balance score and proportional score. Therefore the effectiveness of the approach on the basis of balance is 90.7%. Moreover, 52 out of 54 subjects have lost proportion value. The effectiveness on the basis of proportion score is 96.2%. It confirms that the repaired versions have performed better than the previous mobile friendly technique. Moreover, A survey based evaluation shows that 88% subjects, out of 54, have been considered as more preferable than the previous version by the users.

6.2 Threats to validity

The limitations of the experimental analysis are presented in this section entitled under internal, external and construct validity.

6.2.1 Internal Validity

There is a potential threat in survey based study. The participants might not use full resolution to view on small screen devices. It can impact the survey result. Another threat is the case of dynamic websites. Sometimes, the website content changes dynamically during rendering. I downloaded the subject to use them offline so that no dynamic changes can occur to mitigate the potential threat.

6.2.2 External Validity

There is a bias in the participants selection for the user based survey. I had selected 25 software engineers. I did not specify the qualification level of the participants is the survey. The second threat is the selection of subject websites. I have chosen randomly 64 subjects which ranks are ranged in 1 to 200 from different categories.

6.2.3 Construct Validity

A potential problem is that, the scores given by the participants in response to visual attractiveness are subjective. I had used relative values to handle this threat. Second potential threat is the user based survey is conducted by given screenshots of mobile friendly version and symmetric mobile friendly version of the subjects. I had not any control over the device of the participants. This could cause variation of result.

6.3 Future work

I introduced an approach to get both mobile friendliness and symmetry in a web page. This approach depends on offline version of a website because this can not handle the dynamic changes in rendering. My future work is to handle the dynamic changes along with the symmetric mobile friendliness of a web page. Another possible future work is to analyze the relation between symmetry, mobile friendliness and complexity of a web page to quantify this relation.

Appendix A

Available Viewports for Smartphones and Tabs

Table A.1: Mobile Viewports for responsive experiences-1

Device	**Viewport Size**	**Device Resolution**
iPhone SE	375w x 667h	750w x 1334h
iPhone 11 Pro Max	414w x 896h	1242w x 2688h
iPhone 11 Xs Max	414w x 896h	1242w x 2688h
iPhone 11	414w x 896h	828w x 1792h
iPhone 11 Xr	414w x 896h	828w x 1792h
iPhone 11 Pro	375w x 812h	1125w x 2436h
iPhone 11 X	375w x 812h	1125w x 2436h
iPhone 11 Xs	375w x 812h	1125w x 2436h
iPhone X	375w x 812h	1125w x 2436h

Table A.2: Mobile Viewports for responsive experiences-2

Device	**Viewport Size**	**Device Resolution**
iPhone 8 Plus	414w x 736h	1080w x 1920h
iPhone 8	375w x 667h	750w x 1334h
iPhone 7 Plus	414w x 736h	1080w x 1920h
iPhone 7	375w x 667h	750w x 1334h
iPhone 6s Plus	414w x 736h	1080w x 1920h
iPhone 6s	375w x 667h	750w x 1334h
iPhone 6 Plus	414w x 736h	1080w x 1920h
iPhone 6	375w x 667h	750w x 1334h
iPad Pro	1024w x 1366h	2048w x 2732h
iPad Third & Fourth Generation	768w x 1024h	1536w x 2048h
iPad Air 1 & 2	768w x 1024h	1536w x 2048h
iPad Mini	768w x 1024h	768w x 1024h
iPad Mini 2 & 3	768w x 1024h	1536w x 2048h
Nexus 6P	411w x 731h	1440w x 2560h
Nexus 5X	411w x 731h	1080w x 1920h
Google Pixel	411w x 731h	1080w x 1920h
Google Pixel XL	411w x 731h	1440w x 2560h
Google Pixel 2	411w x 731h	1080w x 1920h
Google Pixel 2 XL	411w x 823h	1440w x 2880h
Samsung Galaxy Note 5	480w x 853h	1440w x 2560h
LG G5	480w x 853h	1440w x 2560h
One Plus 3	480w x 853h	1080w x 1920h
Samsung Galaxy S9	360w x 740h	1440w x 2960h
Samsung Galaxy S9+	360w x 740h	1440w x 2960h
Samsung Galaxy S8	360w x 740h	1440w x 2960h
Samsung Galaxy S8+	360w x 740h	1440w x 2960h
Samsung Galaxy S7	360w x 640h	1440w x 2560h
Samsung Galaxy S7 Edge	360w x 640h	1440w x 2560h
Nexus 7 (2013)	600w x 960h	1200w x 1920h
Nexus 9	768w x 1024h	1536w x 2048h
Samsung Galaxy Tab 10	800w x 1280h	800w x 1280h
Chromebook Pixel	1280w x 850h	2560w x 1700h

Bibliography

[1] "emarketer releases updated estimates for us digital users - emarketer." https://www.emarketer.com/Article/eMarketer-Releases-Updated-Estimates-US-Digital-Users/1015275. (Accessed on 09/28/2020).

[2] "Cloud developer tools — cloud developer tools — google cloud." https://cloud.google.com/products/. (Accessed on 09/28/2020).

[3] "Think with google - discover marketing research & digital trends." https://www.thinkwithgoogle.com/. (Accessed on 09/28/2020).

[4] "Desktop vs mobile market share worldwide — statcounter global stats." https://gs.statcounter.com/platform-market-share/desktop-mobile/worldwide/#monthly-201407-201707. (Accessed on 09/28/2020).

[5] "Mobile-friendly test – google search console." https://search.google.com/test/mobile-friendly. (Accessed on 09/28/2020).

[6] "Multi-screen overview - adsense help." https://support.google.com/adsense/answer/6051803?hl=en&visit_id=637368306470692014-1946767540&rd=1. (Accessed on 09/28/2020).

[7] M. Bauerly and Y. Liu, "Effects of symmetry and number of compositional elements on interface and design aesthetics," *Intl. Journal of Human–Computer Interaction*, vol. 24, no. 3, pp. 275–287, 2008.

[8] T. Lavie and N. Tractinsky, "Assessing dimensions of perceived visual aesthetics of web sites," *International journal of human-computer studies*, vol. 60, no. 3, pp. 269–298, 2004.

[9] X. S. Zheng, I. Chakraborty, J. J.-W. Lin, and R. Rauschenberger, "Correlating low-level image statistics with users-rapid aesthetic and affective judgments of web pages," in *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, pp. 1–10, ACM, 2009.

[10] "Bing - mobile friendliness test tool." https://www.bing.com/webmaster/tools/mobile-friendliness. (Accessed on 09/28/2020).

[11] H. Samimi, M. Schäfer, S. Artzi, T. Millstein, F. Tip, and L. Hendren, "Automated repair of html generation errors in php applications using string constraint solving," in *2012 34th International Conference on Software Engineering (ICSE)*, pp. 277–287, IEEE, 2012.

[12] X. Wang, L. Zhang, T. Xie, Y. Xiong, and H. Mei, "Automating presentation changes in dynamic web applications via collaborative hybrid analysis," in *Proceedings of the ACM SIGSOFT 20th International Symposium on the Foundations of Software Engineering*, p. 16, ACM, 2012.

[13] S. Mahajan, N. Abolhassani, P. McMinn, and W. G. Halfond, "Automated repair of mobile friendly problems in web pages," in *Proceedings of the 40th International Conference on Software Engineering*, pp. 140–150, ACM, 2018.

[14] R. Connolly, *Fundamentals of web development.* Pearson Education, 2015.

[15] E. Bell, D. Zhang, E. L. Soong, H. H. Feng, S. S. Zhu, and H. H. Feng, *Fundamentals of Web Applications Using. NET and XML.* Prentice Hall PTR, 2002.

[16] S. Ratnasingham and P. D. Hebert, "Bold: The barcode of life data system (http://www. barcodinglife. org)," *Molecular ecology notes*, vol. 7, no. 3, pp. 355–364, 2007.

[17] J. Marini, *Document object model.* McGraw-Hill, Inc., 2002.

[18] G. Jakobson, "Collaborative web browsing system having document object model element interaction detection," July 1 2014. US Patent 8,769,017.

[19] "Mobile-friendly test tool - search console help." `https://support.google.com/webmasters/answer/6352293`. (Accessed on 09/29/2020).

[20] "Mobile vs. desktop internet usage (latest 2020 data) - broadbandsearch." `https://www.broadbandsearch.net/blog/mobile-desktop-internet-usage-statistics`. (Accessed on 09/29/2020).

[21] J. D. Foley, F. D. Van, A. Van Dam, S. K. Feiner, J. F. Hughes, E. Angel, and J. Hughes, *Computer graphics: principles and practice*, vol. 12110. Addison-Wesley Professional, 1996.

[22] J. J. McConnell, *Computer graphics: theory into practice.* Jones & Bartlett Learning, 2005.

[23] "Popular screen resolutions — media genesis." `https://mediag.com/blog/popular-screen-resolutions-designing-for-all/`. (Accessed on 09/29/2020).

[24] "Create mobile website — converter & builder — bmobilized." `https://bmobilized.com/`. (Accessed on 09/29/2020).

[25] "Mobify — storefront for headless commerce." `https://www.mobify.com/`. (Accessed on 09/29/2020).

[26] M. Harman and B. F. Jones, "Search-based software engineering," *Information and software Technology*, vol. 43, no. 14, pp. 833–839, 2001.

[27] N. Alshahwan and M. Harman, "Automated web application testing using search based software engineering," in *2011 26th IEEE/ACM International Conference on Automated Software Engineering (ASE 2011)*, pp. 3–12, IEEE, 2011.

[28] P. Yadav and P. N. Barwal, "Designing responsive websites using html and css," *international journal of scientific & technology research*, vol. 3, no. 11, pp. 152–155, 2014.

[29] S. Wesley, "Css media queries," in *Pro jQuery in Oracle Application Express*, pp. 193–205, Springer, 2015.

[30] C. Arney, "Symmetry: A very short introduction," *Mathematics and Computer Education*, vol. 48, no. 1, p. 122, 2014.

[31] K. Mainzer, *Symmetry and complexity: The spirit and beauty of nonlinear science*, vol. 51. World Scientific, 2005.

[32] T.-L. Liu, D. Geiger, and A. L. Yuille, "Segmenting by seeking the symmetry axis," in *Proceedings. Fourteenth International Conference on Pattern Recognition (Cat. No. 98EX170)*, vol. 2, pp. 994–998, IEEE, 1998.

[33] C. W. Tyler, *Human symmetry perception and its computational analysis.* Psychology Press, 2003.

[34] R. Leikin, A. Berman, and O. Zaslavsky, "Applications of symmetry to problem solving," *International Journal of Mathematical Education in Science and Technology*, vol. 31, no. 6, pp. 799–809, 2000.

[35] T. A. Walsh, P. McMinn, and G. M. Kapfhammer, "Automatic detection of potential layout faults following changes to responsive web pages (n)," in *2015 30th IEEE/ACM International Conference on Automated Software Engineering (ASE)*, pp. 709–714, IEEE, 2015.

[36] P. Panchekha and E. Torlak, "Automated reasoning for web page layout," *ACM SIGPLAN Notices*, vol. 51, no. 10, pp. 181–194, 2016.

[37] S. Mahajan, A. Alameer, P. McMinn, and W. G. Halfond, "Automated repair of layout cross browser issues using search-based techniques," in *Proceedings of the 26th ACM SIGSOFT International Symposium on Software Testing and Analysis*, pp. 249–260, ACM, 2017.

[38] A. Alameer, S. Mahajan, and W. G. Halfond, "Detecting and localizing internationalization presentation failures in web applications," in *2016 IEEE International Conference on Software Testing, Verification and Validation (ICST)*, pp. 202–212, IEEE, 2016.

[39] A. Sanoja and S. Gançarski, "Block-o-matic: A web page segmentation framework," in *2014 international conference on multimedia computing and systems (ICMCS)*, pp. 595–600, IEEE, 2014.

[40] D. Chakrabarti, R. Kumar, and K. Punera, "A graph-theoretic approach to webpage segmentation," in *Proceedings of the 17th international conference on World Wide Web*, pp. 377–386, 2008.

[41] N. Bansal, A. Blum, and S. Chawla, "Correlation clustering," *Machine learning*, vol. 56, no. 1-3, pp. 89–113, 2004.

[42] D. Cai, S. Yu, J.-R. Wen, and W.-Y. Ma, "Vips: a vision-based page segmentation algorithm," 2003.

[43] D. Chaudhuri and A. Samal, "A simple method for fitting of bounding rectangle to closed regions," *Pattern recognition*, vol. 40, no. 7, pp. 1981–1989, 2007.

[44] "Eclipse photon — the eclipse foundation." `https://www.eclipse.org/photon/`. (Accessed on 09/30/2020).

[45] "radkovo/jstyleparser." `https://github.com/radkovo/jStyleParser`. (Accessed on 09/30/2020).

[46] "The top 500 most popular websites - moz." `https://moz.com/top500`. (Accessed on 09/30/2020).

[47] "Httrack website copier - free software offline browser (gnu gpl)." `https://www.httrack.com/`. (Accessed on 09/30/2020).

[48] R. Woolson, "Wilcoxon signed-rank test," *Wiley encyclopedia of clinical trials*, pp. 1–3, 2007.